Energy, Mines and Resources Canada | Énergie, Mines et Ressources Canada

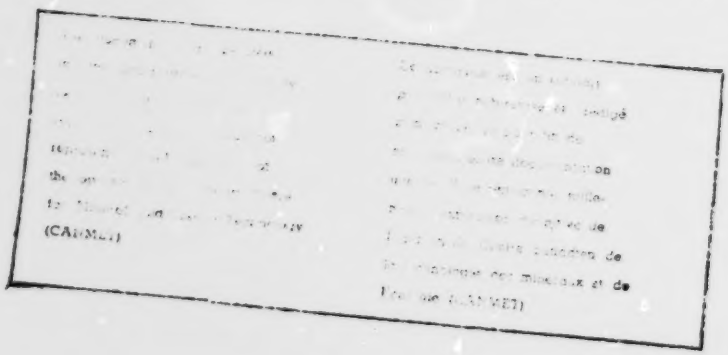
CANMET

Canada Centre for Mineral and Energy Technology | Centre canadien de la technologie des minéraux et de l'énergie

DIGIT PROGRAM SOFTWARE

V.Srajer, N.J.Stuart, C. Neumiller, M.Heximer, W.Veldhuis

March 1987



MICROMEDIA

16 pp

ENERGY RESEARCH PROGRAM
COAL RESEARCH LABORATORIES
DIVISION REPORT ERP/CRL 87-18 (TR)

87-05148/

DIGIT PROGRAM SOFTWARE

by

V. Srajer*, N.J. Stuart**,C. Neumiller***, M. Heximer***, W. Veldhuis***

ABSTRACT

The DIGIT program software presented here is an update of a program written previously. The changes to the original program are listed. The reader is advised to be familiar with the original work before proceeding with this update.

The modified program is now easy to use and less apt to fail due to user error. There were changes made to the screen prompts, input of data, output of data, and internal handling of cases.

---

* Head, Operations Research, CANMET, CRL, Surface Mining Lab., Devon, Alb.
** Research Scientist, CANMET, CRL, Surface Mining Lab., Devon, Alberta.
*** Co-Op Students, University of Alberta, Edmonton, Alberta.

PROGRAMME "DIGIT"

par

V. Strajet*, N.J. Stuart**, C. Neumiller***, M. Heximer***, W. Velhuis***

RÉSUMÉ

Le programme DIGIT décrit dans le présent rapport est une mise à jour d'un ancien programme. Les changements apportés au programme original sont énumérés. Il est préférable que le lecteur connaisse le travail original avant de se servir de cette mise à jour.

Le programme modifié est maintenant facile à utiliser et les échecs dûs à une erreur de l'utilisateur sont moins probables. Des modifications ont été apportées aux messages guide-opérateur, à l'entrée des données, à la sortie des données et au traitement interne des cas.

---

*Directeur, Recherche opérationnelle, CANMET, LRC, Lab. d'exploitation en surface, Devon (Alberta).
**Chercheur, CANMET, LRC, Lab. d'exploitation en surface, Devon (Alberta).
***Étudiants inscrits à un programme de coopération, Université de l'Alberta, Edmonton (Alberta).

CONTENT

Page

ABSTRACT..... i
RESUME..... iii
INTRODUCTION..... 1
PROGRAM DIGIT..... 1
    User Interface..... 1
    Addition of New Segments..... 3
    Input File..... 4
    Output File..... 5
    Notes on Other Changes..... 5
    Program Listing..... 7
CONCLUSION..... 12
REFERENCES..... 12

LIST OF FIGURES

Page

1. Flow Chart of Programs..... 6

INTRODUCTION

The presented DIGIT program is a revised version of the program written previously (1). It is intended to be used to digitize mine haulage roads, provide and store information on the roads digitized, and an add or erase segments of a road. System may be already stored, whether connecting nodes are present or not. The input for this program is a mine layout; output may be plotted on a Calcomp plotter. The layout may use imperial or metric units. The line menu is in simple, easy-to-read format. The position of text to be displayed in the output may be added via the keyboard.

The DIGIT output file is used as an input file by MAP, DAT-VST and DAT-REGIS programs. This file is described in the original work (1).

PROGRAM DIGIT

User Interface.

The program described may be run on any terminal connected to the Calcomp digitizer, using the following procedure:

1. Type RUN DIGIT in directory [IAN.FALL86.MINES].
2. You are asked to "ENTER DATA FILE NAME". Enter the name of the mine site to be digitized.

The following prompts appear on the screen to be considered by the user:

3. "ENTER MINE TITLE". This prompt assigns the name of the mine to the output file. The prompt in 2 above opened the output file only.

4. "ARE UNITS IN FEET ? (Y/N)". If the response is Y, the numbers will be converted to metric units.

Entries 1 to 4 are entered via the keyboard, pressing <CR> after each entry. Most of the remaining entries are either initialized or entered by the digitizer's mouse.

5. "DIGITIZE ORIGIN, USE ANY BUTTON:". The origin is the lover left point of the mine layout. To digitize a point, press a button on the digitizer mouse when the cross-hairs are directly over the point.

6. "DIGITIZE THE X-MAXIMUM, USE ANY BUTTON:". This is any point on the right hand margin of the mine layout.

7. "TYPE IN STARTING AND ENDING VALUES OF X-SCALE:". These are the beginning and ending values on the x-axis, usually given on the layout. They must be entered from the keyboard with a space or comma between the values and each value must contain decimal point.

8. "DIGITIZE THE Y-MAXIMUM, USE ANY BUTTON:". This is the upper bound

of the layout, usually located on the left upper margin.

9. "TYPE IN STARTING AND ENDING VALUES OF Y-SCALE:". Use the same procedure as for the x-axis, only using the y-values.

10. "DIGITIZE TWO POINTS TO FORM A NORTH ARROW, USE ANY BUTTON:". Digitize the tail of the arrow first, so that the arrow will point to the North.

11. "TYPE IN NAME OF THE UNITS:". These are the units associated with the x and y values.

12. "TYPE IN NAME OF THE ELEVATION UNITS:".

13. "TYPE IN NAME OF ROLLING RESISTANCE UNITS:".

14. "TYPE IN NAME OF SPEED RESTRICTION UNITS:".

The compilation of the header data is completed at this point. The program now begins the digitizing of the mine haulage system, segment by segment. A menu appears on the screen as follows:

<A> Starting point on a line.
<B> Point on line, no data.
<C> Point on line, additional data.
<D> End point on line.
<#> End point on final line.

15. To start a line, press the <A> button on the mouse, over the intersection.

16. "ENTER LINE NAME:". Type in the segment linename. It must be of the format NODE1-NODE2, N1-N2, etc., with the minus sign separating the node or intersection names.

17. "ENTER ROLLING RESISTANCE:". The rolling resistance for the point is entered (eg. XX.XX). The rolling resistance, elevation and speed restriction data are required to be entered at the starting and ending points of each segment. The data must be entered in the format shown.

18. "ENTER ELEVATION (eg. XXXXX.XXX)".

19. "ENTER SPEED RESTRICTION (eg. XXX.X)". Once the data has been entered, the menu reapears.

20. An intermediate point without any accompanying data is digitized by pressing the <B> button. The menu is then shown once more.

21. A middle point that is followed by additional data is digitized by pressing the <C> button. The prompts for the rolling resistance, elevation and speed restriction data appear in the order described previously. If any of the data remains unchanged, simply press the <CR> key. Once all three data types have been entered, the menu is displayed again.

22. To end a segment which is not the last segment in the layout to be digitized, the <D> button is pressed over the last point of the segment. Data for all three types must be entered for the point, after which the menu is again listed.

23. The completion of the segment digitizing is accomplished by pressing the <#> key over the last point of the last segment on the layout. Similarly, as with the <D> button , rolling resistance, elevation and speed restrictions data must be entered.

The entering of the data is now complete. Any text which has to be added to the file is entered now. The text may include specific areas or names, such as shop, maintenance bay, etc, which are to be displayed on plots or mine displays.

24. If no text is to be entered, press the <#> button.

25. "DIGITIZE THE NEXT POSITION USE <A> (<#> TO EXIT):". The starting position of the text is entered (lower left of the first letter).

26. "ENTER TEXT (FROM THE KEYBOARD):".

27. "ENTER ANGLE OF TEXT (FROM KEYBOARD):". The angle is entered in degrees.

28. "ENTER THE SIZE OF THE TEXT (FROM KEYBOARD):". The units are cm.

29. The above procedure is repeated until all of the text has been entered.

30. Press the <#> button to end the program.

Addition of New Segments.

If new segments are to be added to the DIGIT output file, ne segments of the haulage system may be added to the existing file. There are two methods of including these segments. The first method is used when a new segment is to be added onto an existing node, the second method is used to add new segments to a haulage system when no connecting node exists.

First method:

1. Run DIGIT.

2. Use 'APPEND' as a file name.

3. Use the same header data that was used during original digitizing. Care must be taken to have the origin in the same place.

4. Digitize new segment(s).

5. Digitize any additional text.

6. End digitizing.

7. "Append APPEND.DAT <MINE>. DAT". This $ command will attach the append data file to the mine data file.

8. Edit the <MINE>.DAT file.

9. Search for the first occurence of '#' that appears at the left hand margin of a line. Change the '#' to a 'D'.

10. Move to the first text line. The text lines are moved to the end of the file by doing the following:

    -press the <.> key on the key pad located to the right of the keyboard,
    -move the cursor until all of the text lines are highlighted,
    -press the <6> key,
    -move the cursor to end of file, or press the <PF1> key, then the <4> key, and
    -press the <PF1> key, then the <6> key.

11. Search for the 'FILENAME' located in the file (not the name located at the beginning of the file).

12. Place the cursor at the beginning of this line.

13. Press the <PF4> key nine times.

14. Press the <PF1> key, the <4> key, the <6> key.

Second method:

1. Repeat steps 1 to 3 above.

2. Re-digitize the segment that the new segment will attach to.

3. Repeat steps 4 to 8 from above.

4. Search for the old segment that was re-digitized. Delete the entire segment so that it will not interfere with the new segment and node.

5. Repeat steps 9 to 14 from the first method.

Input File.

The input for this program is the mine layout of the haulage roads, pit system, engineering drawings, etc.

Output File.

The DIGIT output file is signified by the mine name (FILENAME) followed by the .DAT suffix. eg. HARMER.DAT. The file itself consists of three parts:

- header data,
- line data, and
- text data.

The header data includes the filename, x and y value ranges, the north arrow's point co-ordinates, and the units of the plot, elevation, rolling resistance and speed restriction.

The line data is separated from the header data by a blank line, each recent line is separated from each other by the same. The line data includes the line name, all points digitized on the line segment and any additional data given for each point. The point co-ordinates are given first. This line begins with the letter that signifies if it is the first, middle or last point of the segment: an 'A' starts a segment, a 'B' or 'C' is a middle point, and a 'D' or '#' is an end point. Additional data lines are started with an ampersand (&) if the line is to be followed by another data line, or a '%' sign when the next line will be a co-ordinate line. Each point on a segment must have its own (x,y) co-ordinates, as well as the beginning and ending points of a segment containing values for all three input parameters.

The text data includes the text itself, the angle of the text (in degrees), the text size (in cm) and the text starting co-ordinates, in that order.

The DIGIT output file is used as an input file by other programs, described in separate reports. The MAP program uses the file to plot the mine haulage system on the Calcomp plotter and to output the distance of each segment. The DAT_REGIS program converts the file to a structure that is required by the RMAP program, changing it to a .REG file from a .DAT file. The DAT_VST program also converts the file to a structure that is required by the VMAP program, changing it to a .VST file.

Notes on Other Changes.

The following are revisions that were made to the DIGIT program described in (1).

1. The addition of a prompt for a mine title.

2. The addition of unit check prompt (feet or metres).

3. The simplifying of the digitizing of the header data.

4. The assigning of the origin values in the program instead of calling a subroutine to perform the task.

5. Conversion of x and y scale values to metric if entered in imperial

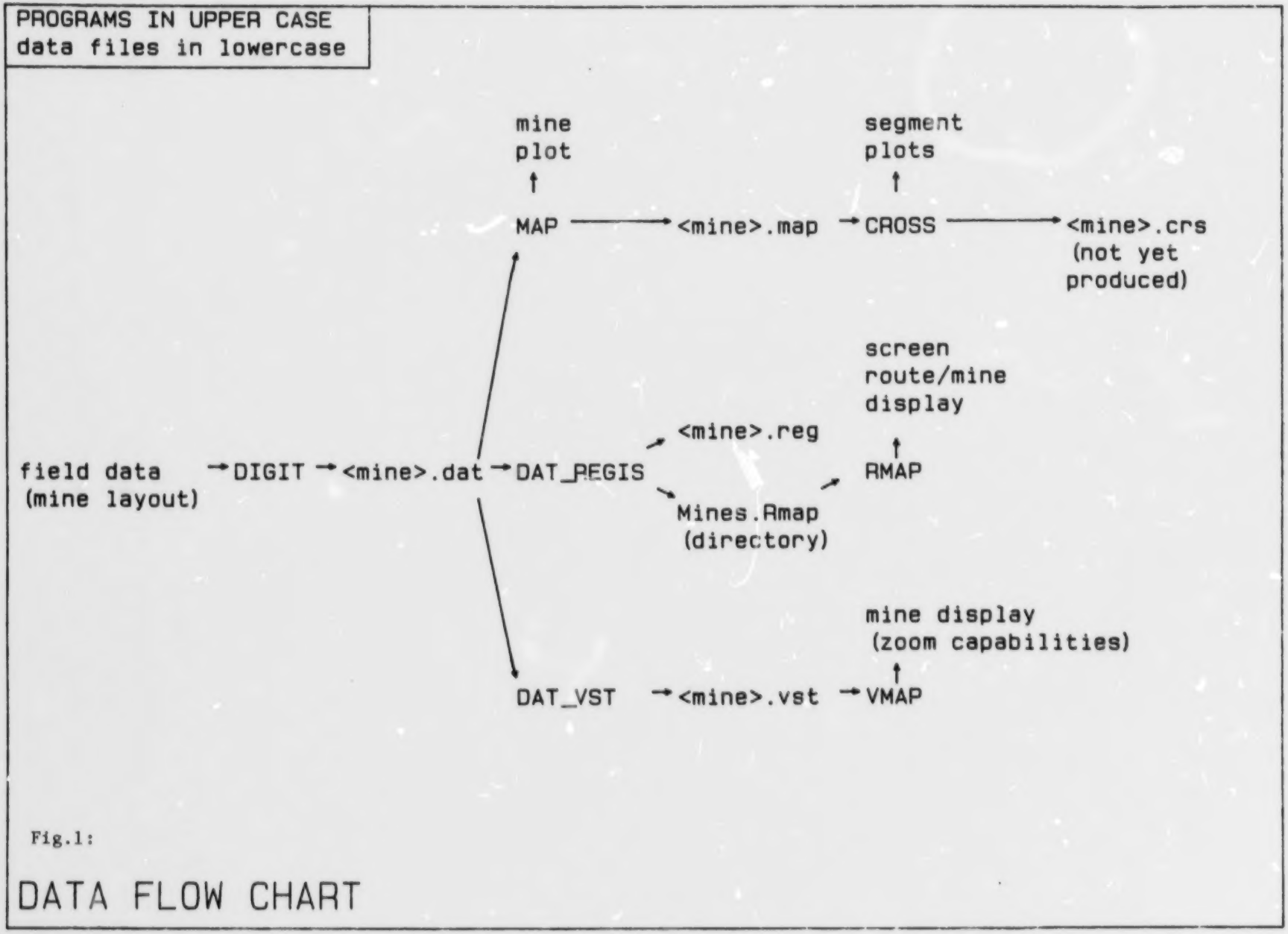


Fig.1:

DATA FLOW CHART

units.

6. The line menu was changed to an easier-to-read format.

7. The procedure needed to enter any additional data when it is called for was changed.

8. A section was added to digitize the text position, then enter the text, size and angle, the last three via the keyboard.

Program Listing.

The flow chart of the programs and data files mentionned in the report is shown in the enclosed Figure 1.

The listing of DIGIT program written in VAX FORTRAN 77 is as follows:

```
      PROGRAM DIGIT

c     LINK DIGIT
c     PROGRAM THAT READS DATA FROM THE DIGITIZER INTO A DATA FILE

      logical flag
      real x,y,xo,yo,startx,starty,factx,facty,endx,endy,zdata,rr,el,sr
      real size
      integer n
      character*30 filename,linename,units,elunits,ty,addat
      character*30 srunits,rrunits
      character*1 but,dum

c     x,y             = data points
c     n               = # of points digitized, including the origin
c     filename        = name of data file
c     but             = first and only needed element of button
c     xo,yo           = co-ordinates of the origin
c     linename        = name of line
c     zdata           = additional data that is required
c     endx,endy       = last value for scale
c     startx,starty   = first value for scale
c     units           = units used for data
c     elunits         = elevation units used
c     srunits         = speed restriction units
c     rrunits         = rolling resistance units
c     factx,facty     = scale factor
c     ty              = type of additional data entered
c     addat           = determines where to move within program
c                       after additional data has been entered
c     size            = text size
c     dum             = dummy variable
c     flag            = logical check for units

c     NAME DATA FILE

      type*,'enter data file name, press return <CR>'
      read(5,10),filename
10    format(a)
```

```
c       OPEN DATA FILE

        open(unit=1,file=filename,status='unknown')

c       WRITE FILENAME TO FILE

        write(*,*) ' Enter mine title :'
        read(*,'(a)') filename
        write(1,*),'filename:  ',filename

c       CHECK UNITS

        write(*,*) ' Are units in feet? (Y/N)'
        read(*,'(a1)') dum
        flag=.false.
        if(dum.eq.'Y'.or.dum.eq.'y') flag=.true.

c       READ IN DATA

c       DEFINE ORIGIN

        n=2
        factx=1
        facty=1
        type*,' Digitize origin, use any button :'
        read(5,20,end=1000),but,dum,x,y

        xo=x
        yo=y

c       DETERMINE X-MAXIMUM

        type*,' Digitize the x-maximum, use any button :'
        read(5,20),but,dum,x,y
        call calc(n,x,y,xo,yo,factx,facty,startx,starty)

c       DETERMINE X SCALE VALUES

        type*,'Type in starting and ending values of x-scale :'
        read(5,*),startx,endx
        if(flag) then
                startx=tomet(startx)
                endx=tomet(endx)
        endif
        write(1,*)'x-value range:  ',startx,' - ',endx
        call scale(x,startx,endx,factx)

c       DETERMINE Y-MAXIMUM

        type*,' Digitize the y-maximum, use any button :'
        read(5,20),but,dum,x,y
        call calc(n,x,y,xo,yo,factx,facty,startx,starty)

c       DETERMINE Y SCALE VALUES

        type*,'Type in starting and ending values of y-scale :'
        read(5,*),starty,endy
        if(flag) then
                starty=tomet(starty)
                endy=tomet(endy)
        endif
        write(1,*),'y-value range:  ',starty,' - ',endy
        call scale(y,starty,endy,facty)
```

```
c       DETERMINE WHICH DIRECTION IS NORTH

        type*,'  Digitize 2 points to form a north arrow, use any button :'
        read(5,20),but,dum,x,y
        call calc(n,x,y,xo,yo,factx,facty,startx,starty)
        write(1,'(2f11.3)') x,y
        read(5,20),but,dum,x,y
        call calc(n,x,y,xo,yo,factx,facty,startx,starty)
        write(1,'(2f11.3)') x,y

c       DEFINE VARIABLE UNITS

        type*,'Type in name of the units :'
        read(5,10),units
        write(1,*),'plot units:  ',units

        type*,'Type in name of the elevation units :'
        read(5,10),elunits
        write(1,*),'elevation units:  ',elunits

        type*,'Type in name of the rolling resistance units :'
        read(5,10),rrunits
        write(1,*),'rolling resistance units:  ',rrunits

        type*,'Type in name of the speed restriction units :'
        read(5,10),srunits
        write(1,*),'speed restriction units:  ',srunits

c       DIGITIZE EACH SEGMENT

77      type *,'<A> Starting pt. on line'
        type *,'<B> Pt. on line, no data'
        type *,'<C> Pt. on line, additional data'
        type *,'<D> Ending pt. on line'
        type *,'<#> Ending pt. on final line'

20      format(2a1,f6.2,f6.2)

        read(5,20),but,dum,x,y
        call calc(n,x,y,xo,yo,factx,facty,startx,starty)

c       GET LINE NAME

        if(but.eq.'A')then
            type*,'Enter line name :'
            read(5,10),linename
            write(1,*)
            write(1,40),linename
40          format(1x,a)
        endif

c       WRITE DATA TO FILE

        write(1,50),but,x,y
50      format(1x,a1,f9.2,f9.2)

        if(but.eq.'B') goto 77

c       WRITE ROLLING RESISTANCE, ELEVATION, SPEED RESTRICTION TO FILE

60      format(2x,a1,2x,a2,1x,f10.3)
        type*, ' Enter rolling resitance (eg. XX.XX)'
        read(5,'(f5.2)') rr
```

```
      type* , 'Enter elevation          (eg. XXXXX.XXX)'
      read(5,'(f9.3)') el
      type* , 'Enter speed restriction (eg. XXX.X)'
      read(5,'(f5.1)') sr
      if(rr.ne.0.0) then
            dum='&'
            if(el.eq.0.0.and.sr.eq.0.0) dum='%'
            write(1,60) dum,'RR',rr
      endif
      if(el.ne.0.0) then
            dum='&'
            if(sr.eq.0.0) dum='%'
            if(flag) el=tomet(el)
            write(1,60) dum,'EL',el
      endif
      if(sr.ne.0.0) write(1,60) '%','SR',sr

c     EXIT DIGITIZING LINES

      if(but.eq.'#') goto 900

      goto 77

c     DIGITIZE TEXT

900   write(*,*) 'Digitize text position use <A>,(<#> to exit) :'
      read(5,20),but,dum,x,y

      if(but.ne.'A'.and.but.ne.'#') goto 900

      call calc(n,x,y,xo,yo,factx,facty,startx,starty)

      if(but.ne.'#') then
        write(*,*) ' Enter text, start with a space (from keyboard) :'
        read(*,'(a20)') filename
        write(*,*) ' Enter angle of text (from keyboard) :'
        read(*,*) angle
        write(*,*) ' Enter size of text (from keyboard) :'
        read(*,*) size
        write(1,'(a20,f4.0,f5.2,2f9.2)') filename,angle,size,x,y
        goto 900
      endif

c     CLOSE DATA FILE

1000  close(unit=1)

      stop
      end

**********************************************************************

      SUBROUTINE CALC(n,x,y,xo,yo,factx,facty,startx,starty)

c     A SUBROUTINE TO CALCULATE DATA POINTS WITH REGARD TO THE ORIGIN

      real x,y,xo,yo,factx,facty,startx,starty
      integer n

c     n              = number of times calc subroutine called
c     x,y            = data points
c     xo,yo          = co-ordinates of uncrrected origin
c     factx,facty    = correct value of each point
c     startx,starty  = initial values of x and y
```

```
c       DEFINE ORIGIN

        if(n.eq.1)then
            xo=x
            yo=y
        endif

        n=n+1

c       SUBTRACT ORIGIN FROM DATA POINTS

        x=x-xo
        y=y-yo

c       FIND TRUE VALUE OF POINTS

        if(n.gt.3)then
            x=(x*factx)+startx
            y=(y*facty)+starty
        endif

c       RETURN TO MAIN PROGRAM

        return
        end

*****************************************************************

        SUBROUTINE SCALE(x,start,end,fact)

c       SUBROUTINE TO CALCULATE SCALING FACTORS

        real x,start,end,fact


c       x     = data point
c       star  = first point along scale
c       end   = last point along scale
c       fact  = scale factor

        fact=(end-start)/x

c       RETURN TO MAIN PROGRAM

        return
        end

*****************************************************************

        FUNCTION TOMET(imp)

c       A FUNCTION TO CONVERT FEET TO METRES

        real imp

        tomet=imp*.3048

c       RETURN TO MAIN PROGRAM

        return
        end
```

CONCLUSION

A complex set of the programs were prepared and tested to be used when mine haulage roads are to be digitized. The program is providing aid to the professional investigating mine haulage profiles, rolling resistances and speed restrictions on various segments of the road. It will help in identification of alternatives during mine haulage system optimization studies.

The program DIGIT was documented in this form to provide a user-friendly guide to the mine engineer with limited computer background.

REFERENCES

1. C. Neumiller, N.J. Stuart, M. Heximer, W. Veldhuis : "Digitizing Programs to Generate Input Data for a Haulage Simulation Model by Using Fitting Cubic Curves", ERP/CRL Divissional Report 86-129 (TR), CANMET, CRL, Devon, Alberta.